CU Launch

First-Time User Guide

Launch, publish, manage, and troubleshoot Cedarville apps without prior development or hosting experience.

Includes the complete guide, troubleshooting, FAQ, and glossary.

CU Launch User Guide

This guide explains the complete app lifecycle in plain language. Start with the Quick Start if you only need the shortest path to a first published app.

1. What the portal does

CU Launch brings the main pieces of an app into one managed workflow. It can create starter code from an approved template, keep that code in a managed GitHub repository, prepare Azure hosting, publish the app, and help you manage access afterward.

The portal does not design every screen or write every business rule for you. After creating the starter, you can work with Codex or a developer to customize the app. GitHub remains the supported source of truth: the version stored there is the version Cedarville tools review and publish.

2. Understand the app lifecycle

1. **Launch or add:** Select **Launch New App** for a template or **Add Existing App** for code that already exists.
2. **Customize:** Use Codex or a developer to change the managed GitHub repository.
3. **Prepare:** The portal adds and checks the settings needed for GitHub and Azure to work together.
4. **Publish:** The portal starts the GitHub workflow that sends the current app to Azure.
5. **Manage:** Before the first successful publish, use **Continue Setup**. After success, use **Manage App** for the full details and management controls.

3. Choose the right starting point

Launch New App

Choose **Launch New App** when you are starting a new project and want Cedarville-approved defaults. The entry page first explains that choosing a template is the next step. Recommended Templates are written for common, non-technical use cases. Developer Starters expose lower-level choices and are better when a developer already knows the intended architecture.

Common template choices include:

- A PostgreSQL database for information the app must save.
- An audience choice: require Cedarville sign-in or make the app openly public.
- A web interface for forms, trackers, and information pages.
- An API or automation service for a system-to-system process without a normal web page.

Choose only the options your app needs. The portal no longer asks you to choose a runtime; it prepares that technical detail for you.

Add Existing App

Choose **Add Existing App** when code already exists.

- **Already on GitHub:** Give the portal the repository address. If necessary, the portal imports it into Cedarville's managed GitHub organization while preserving its history. The portal checks whether it matches a supported Azure App Service runtime.

- **Only on my computer:** Let the portal create an empty managed repository. The focused wizard provides a prompt that makes Codex responsible for checking the folder, protecting secrets, preserving existing change history, connecting the managed repository, and uploading the code.

Imported apps currently support root Next.js, Express, Python FastAPI, and plain static Python apps. A repository with conflicting publishing files may require a GitHub review page before the portal applies its setup.

4. Launch and choose what happens next

1 From the home page, select **Launch New App** and then **Choose an app template**.
2 Read the template summaries and select the closest match.
3 Enter a short, recognizable app name. Avoid department abbreviations that coworkers may not understand.
4 Describe the app's purpose and intended users.
5 Choose who can use the app:

- **Cedarville sign-in required** means users must sign in with a Cedarville account.
- **Openly public on the internet** means anyone who knows or discovers the app address can use it. Read and confirm the warning before choosing this option.

6 Review the optional database choice.
7 Select **Launch App**. Creation stops before publishing.

When **Your starter app is ready** appears, choose **Publish the starter now** or **Customize it with Codex first**.

Publish the starter now starts Azure publishing immediately. It is the only publish confirmation for an unchanged starter and does not require your own GitHub account. **Customize it with Codex first** starts the GitHub access and Codex handoff steps. Codex is optional; after customized code is pushed and setup is ready, that path later presents **Publish to Azure**.

5. GitHub access and Codex customization

GitHub is the managed online location for the app's code. It records changes and lets Cedarville's publishing workflow use a reviewed source.

Git is the tool that keeps change history in the app folder on your computer. Before opening a Codex handoff:

1 On Windows, open **Company Portal**, search for **Git**, and select Install. On macOS, open **CedarNet 2.0**, search for **Git**, and select Install.
2 When installation finishes, completely quit and reopen Codex.
3 Make a new empty folder named for a generated app. For an app already on your computer, keep using the folder that already contains it.
4 In Codex, open Projects and create a **local Codex project** using that folder. Make it the primary folder, which tells Codex where the app's files belong.
5 Open that project and start the task inside it. Do not use Quick chat or start a standalone task outside the project.
6 Paste the portal's complete prompt into that project task. Codex checks the folder and Git before changing anything.

If you choose customization, the wizard asks whether you already have a GitHub account. A GitHub account is a login for the website where the app's private code home is stored. To continue:

1 If you need an account, open the supplied GitHub signup link. Choose a GitHub username during signup, complete GitHub's verification, sign in, and return to the same portal tab.
2 Enter the username you chose and select **Send repository invite**. The username is not your email address or display name. You can find it after `github.com/` in your GitHub profile address.
3 Accept the private repository invitation on GitHub.
4 Return and select **I've accepted the invitation**.

5 Follow the **Before opening Codex** checklist, then copy the complete prompt into the task inside the app's local Codex project. Let Codex inspect, change, test, commit, and push the app.
6 Return only after Codex reports that the push succeeded.

Portal collaboration and GitHub access are separate. Confirm that finished changes are committed and pushed to the managed repository; the portal cannot publish local files that were never pushed. Codex uses an HTTPS repository address and may open a secure browser or operating-system GitHub sign-in. Complete that sign-in yourself. Never provide a GitHub password, personal access token, SSH key, or other secret. Codex should stop and direct you to Cedarville IT if secure sign-in does not work; it should not use the GitHub plugin or GitHub CLI as a fallback.

Permission prompts

Codex may ask to use the app folder, run a normal development command, or open secure GitHub sign-in. Read each request and allow only what is needed for the current app task. If a prompt offers **Allow once**, choose that option unless Cedarville IT has approved an established workflow that needs broader access.

It is normally appropriate to allow access to the app folder you selected and to complete secure browser sign-in yourself. Do not approve access to unrelated folders or applications, and never paste passwords, tokens, private keys, or portal credentials into Codex.

6. Add code that already exists

Already on GitHub

Paste the repository's web address and give the app a recognizable name. The portal makes or reuses the managed Cedarville copy, preserves the source history, and checks whether it matches a supported Azure runtime. It supports root Next.js, Express, Python FastAPI, and plain static Python apps.

Select **Prepare my app for publishing** when offered. The portal adds only the publishing files the app needs. If existing files conflict, it does not overwrite them. Select **Open a safe review on GitHub** to create a pull request, which is a GitHub page showing proposed changes. Review and merge it, then select **I've approved the changes** so the portal can verify them.

Only on my computer

Enter the local app name and select **Create online home**. Follow the GitHub account or invitation steps if shown. Create a local Codex project from the folder that already contains the app, make that folder primary, and start the task inside the project, not Quick chat. Copy the local-upload prompt into that task. Codex preserves your history, connects the managed repository, and pulls the portal's app guidance before changing anything. It then checks whether the app already uses a supported type. If not, it explains and performs the smallest safe migration to Next.js, Express, FastAPI, or a plain static app while preserving the app's behavior. Codex tests the result before pushing it. Select **My code has been uploaded** only after Codex reports success.

The portal then scans and prepares the uploaded code. If the runtime is unsupported, it returns a repair prompt to Codex. After Codex repairs, tests, and uploads the app, select **I've repaired and uploaded my code**. Do not use either confirmation before the upload succeeds.

7. Publishing to Azure

Publishing makes the current GitHub version available as a running website in Azure.

Before publishing, confirm:

- Repository status is ready.
- Publishing setup is ready.

- Required environment variables have been added.
- The code you want is present in the managed GitHub repository.

For a customized generated app or an imported or local app, select **Publish to Azure** from the focused wizard after preparation and setup are ready. This separate button is the explicit confirmation for those paths; setup and repair never publish on their own. An unchanged generated starter does not show a second confirmation because **Publish the starter now** already starts publishing. Publishing may take several minutes. Preparation, setup checks, repairs, and publishing pages refresh automatically, so leave the page open and do not repeat an action while it is running.

When publishing succeeds, the wizard shows **Your app is online**, **Open your app**, and **Open app details**. Open the app first to check its main task. The details page includes a plain-language readiness summary; technical management controls are under **Advanced options**. If you leave earlier, My Apps shows **Continue Setup** and resumes the exact safe step. After success, My Apps shows **Manage App**.

Share in Portal is separate from app access. It lets signed-in Cedarville portal users discover the app's name, description, and link. It does not make an app public or change the audience you chose at creation.

After publishing, open the app and test its most important task. A successful deployment only proves that Azure started the app; it does not prove every form, permission, integration, or data rule behaves correctly.

Updating an app

1. Make and test the change with Codex or a developer.
2. Push the change to the managed GitHub repository.
3. If push-to-deploy is enabled, the GitHub workflow may publish automatically. Otherwise, return through **Manage App** and select **Publish to Azure**.
4. Test the published app again.

8. Environment variables and secrets

Environment variables provide settings the app needs at runtime, such as an external service address or secret credential. Add them from the app details page instead of placing secret values in the code or documentation.

The portal stores user-managed secret values in an app-specific Azure Key Vault and gives only that app access. The portal does not show the secret value again after it is saved.

Good practices:

- Use the exact variable name provided by the app or integration instructions.
- Paste only the value, without explanatory text.
- Replace a variable when its credential rotates.
- Republish or restart as directed after changing a required setting.
- Never put real secrets in GitHub files, screenshots, support tickets, or email.

9. Collaborators and ownership

Every app has one primary owner. Owners and administrators can invite Cedarville coworkers by email. The coworker must accept the invitation through Cedarville sign-in before becoming a collaborator.

Collaborators can view app details, request their own GitHub access, repair publishing setup, and publish changes. They cannot delete app resources or transfer ownership. An administrator can reassign the primary owner when responsibility changes.

Removing a collaborator ends portal access immediately. GitHub access is revoked on a best-effort basis when the portal knows the person's GitHub username; verify repository access separately when removing someone from sensitive work.

10. Recovery and publishing setup

During first setup, use **Fix publishing setup** when the wizard says a protected connection needs attention. On the full details page, the same recovery appears as **Repair Publishing Setup**. Repair refreshes portal-managed GitHub secrets, Azure connection settings, and federated credentials for that app.

Repair does not delete the repository or Azure app, and it does not start a deployment. When repair finishes, select **Publish to Azure** separately if you want to deploy code. Preparation failures use **Try preparation again** with the saved safe method; publishing failures use **Try publishing again**. If a repeated failure has no safe action, use the displayed support reference.

11. Delete an app carefully

Deletion is divided into separate scopes:

Choice	What it removes
Portal record	Removes the app from My Apps.
GitHub repository	Deletes the managed source repository and its history.
Azure deployment	Deletes the app's Azure Web App and its app-specific database when one exists.

Azure deletion does not remove Cedarville's shared App Service Plan or shared PostgreSQL server. If you delete only the portal record and leave GitHub or Azure unchecked, those resources remain but can no longer be managed from My Apps. Arrange manual cleanup with Cedarville IT if that happens.

Before deleting, confirm the app name, decide whether the code or data must be retained, and coordinate with collaborators. Repository and Azure deletion can be difficult or impossible to reverse.

12. Getting help

Start with Troubleshooting, then review the FAQ. If you still need help, contact Cedarville IT using the approved support channel.

Provide the app name, support reference if shown, approximate time, action you selected, exact on-screen message, and a screenshot with sensitive values hidden.

Do not provide passwords, client secrets, database connection strings, environment-variable values, private keys, or downloaded app source unless support specifically arranges a secure transfer.

Troubleshooting

Start with the row that most closely matches what you see. Read the full on-screen message before retrying. Avoid repeatedly selecting an action while it is still running.

For every Codex handoff, first open a **local Codex project** whose primary folder is the app folder. Do not use Quick chat or a standalone task. Git comes from **Company Portal** on Windows or **CedarNet 2.0** on macOS.

What you see	What it usually means	What to do
You cannot sign in	The Cedarville session, account, or portal authentication configuration needs attention.	Close other Microsoft sign-in prompts, use your Cedarville account, and try again. If it continues, contact support.
An app creation form will not submit	A required field is empty or has an unsupported value.	Read the message beside each field, correct it, and submit again.
You are unsure whether an app is public	The app audience controls who can open the published app.	Open App details and read App readiness. **Cedarville sign-in required** limits access to Cedarville accounts; **Openly public** means anyone who knows or discovers the address can use it. **Share in Portal** only lists the app for signed-in portal users.
The code home is still being prepared	The portal is creating or checking the managed repository.	Leave the page open. It checks automatically and moves to the next safe step.
Repository setup failed	GitHub could not create or import the managed repository.	Follow the offered restart or retry once. Confirm the repository address is correct and accessible. Then provide the displayed support reference to support.
A GitHub invitation is pending	GitHub access was requested but has not been accepted.	Sign in to the correct GitHub account, check notifications and email, and accept the repository invitation.
The portal does not know your GitHub username	Your portal profile is missing the account name used on GitHub.	If needed, create the account from the wizard first. Enter the name after `github.com/` in your profile address, not your email address or display name, then save it and return to the app.
Codex says Git is not available	Git is not installed, or Codex was open during installation and cannot see it yet.	Do not let Codex install anything. Install Git from **Company Portal** on Windows or **CedarNet 2.0** on macOS. Completely quit and reopen Codex, reopen the local project, and return to the same task.
Codex is in Quick chat or the wrong folder	The handoff was started outside the app's local project.	Stop without changing files. Create or open the **local Codex project**, make the app folder primary, and start the task inside it. For a generated app, use a new empty folder; for an existing local app, use its current folder.
GitHub sign-in does not open or fails	Git could not complete its secure browser or operating-system sign-in.	Stop and contact Cedarville IT. Do not give Codex a password, personal access token, or SSH key, and do not switch to the GitHub plugin or GitHub CLI.
Codex asks for a permission you do not recognize	The task may be requesting more access than the current app work needs.	Read the request. Use **Allow once** if available for the selected app folder, normal development commands, or secure sign-in you complete yourself. Decline unrelated folders or applications and contact Cedarville IT if unsure.
My code has been uploaded is shown	The portal is waiting for Codex to finish the local upload.	Do not select it early. Wait for Codex to report a successful push, then select it so the portal can inspect the uploaded code.
The local app cannot be prepared	The uploaded runtime is not supported yet, or the earlier compatibility check missed a required change.	Copy the repair prompt into the same local Codex project. Codex uses the pulled portal skill to choose the smallest safe migration, preserve the app's behavior, test it, and push it. Then select **I've repaired and uploaded my code**.

What you see	What it usually means	What to do
Preparation needs another try	The portal could not finish the saved preparation method.	Select **Try preparation again** once. The portal reuses the same safe direct-update or review method.
A safe GitHub review is offered	Existing publishing files overlap with the portal's proposed files.	Select **Open a safe review on GitHub**, review and merge the pull request, then select **I've approved the changes**. Ask a developer if you do not recognize the files.
Publishing setup needs attention	A managed credential or required setting is stale or missing.	Select **Fix publishing setup** in onboarding or **Repair Publishing Setup** in full details. When setup is ready, select **Publish to Azure** separately.
Publish failed	GitHub Actions, Azure, the app build, or a required setting failed.	Open the failure details if available. Select **Try publishing again** once. If offered after a repeated problem, fix publishing setup before another try.
Published app shows an error page	Azure started a deployment, but the app may have a code, startup, database, or configuration error.	Confirm required environment variables, then republish. Record the time and error text for support.
Your latest change is missing	The change may not have been pushed to the managed repository or republished.	Confirm the commit is on GitHub, publish again if needed, then refresh the app without using the browser cache.
A collaborator cannot open the app in the portal	The invitation may not be accepted or the wrong Cedarville address was used.	Ask the person to use the invitation link and Cedarville sign-in. Remove and resend the invitation if the address is wrong.
A collaborator cannot open GitHub	Portal collaboration does not automatically guarantee GitHub access.	Have the collaborator save their GitHub username and request repository access from the app page.
An environment variable did not fix the app	The name may be wrong, the value may need rotation, or the app may need another publish.	Verify the exact name, replace the value, and follow the app's instructions to republish. Never send the value to support.
A delete option is unavailable	Your role or the current resource state does not allow that deletion.	Confirm you are the primary owner or ask a portal administrator for help.

Safe retry sequence

1. Wait for the current action to finish or clearly fail.
2. Read and save the exact message and support reference.
3. Refresh the page once.
4. Use **Fix publishing setup** or **Repair Publishing Setup** only when the setup status calls for it.
5. Retry the failed action once.

If the same problem repeats, stop and contact support.

What to send support

- App name and support reference.
- The approximate date and time.
- The page and button you used.
- Exact message or a screenshot with sensitive information hidden.
- Whether the issue happened once or repeated after one retry.

Never send secret values, passwords, private keys, database connection strings, or the contents of environment variables.

Frequently Asked Questions

Do I need to know how to program?

No. Recommended Templates and the portal workflow are designed for first-time users. You may still work with Codex or a developer to customize behavior beyond the starter.

What is GitHub, and why does the portal use it?

GitHub is a managed online place for app code and change history. It gives Codex, reviewers, collaborators, and Azure publishing one supported source for the app.

What is Azure?

Azure is Microsoft's cloud platform. The portal uses Azure App Service to run each published web app. Apps may also receive an app-specific database and secure secret storage when required.

Which template should I choose?

Choose the Recommended Template whose examples most closely resemble your project. Use a Developer Starter only when a developer has identified the needed runtime or service type.

Can I change templates later?

Not as a simple switch. A template shapes the starter code and available features. Ask Codex or a developer whether adapting the current app or creating a new one is safer.

Does publishing make my app public?

No. Publishing makes the app run in Azure. At creation, choose either **Cedarville sign-in required** or **Openly public on the internet**. An openly public app can be used by anyone who knows or discovers its address. Cedarville sign-in limits entry to people who can sign in with their Cedarville account.

Is Share in Portal the same as making my app public?

No. **Share in Portal** lets signed-in Cedarville portal users see the app's name, description, and link. It does not change who can open the published app.

Does creating a starter publish it?

No. **Launch App** makes the starter and its managed repository, then stops. **Publish the starter now** starts Azure publishing immediately and is the only publish confirmation for an unchanged starter. **Customize it with Codex first** leads through GitHub access and setup before a later **Publish to Azure** action.

Do I need a GitHub account?

Not to publish an unchanged generated starter. You need one when Codex or a person must access the private repository. If you do not have one, the wizard opens GitHub signup and explains how to choose a username. After signup, enter that username - not your email address or display name. It appears after `github.com/` in your profile address. The portal then sends and confirms the repository invitation.

What do I need before using Codex?

Install Git from **Company Portal** on Windows or **CedarNet 2.0** on macOS, then completely quit and reopen Codex. For a generated app, make a new empty folder. For an app already on your computer, use its existing folder. In Codex, create a **local Codex project** from that folder and make it primary. Start the task inside the project; do not use Quick chat or a standalone task. The portal prompt handles Git commands and uses secure browser sign-in for GitHub. Never provide a password, personal access token, or SSH key.

What should I do when Codex asks for permission?

Read the request and allow only what the current app task needs. It is normal to allow the selected app folder, normal development commands, and secure browser sign-in that you complete yourself. Choose **Allow once** when it is available, unless Cedarville IT has approved a broader established workflow. Do not approve unrelated folders or applications, and never provide passwords, tokens, private keys, or portal credentials.

What is the difference between Continue Setup and Manage App?

Continue Setup returns an unpublished app to its focused first-publish wizard. **Manage App** appears after a successful first publish and opens the full app details page.

What happens after someone changes the code?

The change must be committed and pushed to the portal-managed GitHub repository. Then push-to-deploy may publish automatically, or an authorized portal user can select **Publish to Azure**.

Can another person help manage my app?

Yes. The owner or an administrator can invite a Cedarville coworker as a collaborator. The collaborator accepts through Cedarville sign-in and requests separate GitHub access when needed.

What does Repair Publishing Setup do?

It refreshes portal-managed GitHub and Azure publishing credentials and settings. It does not delete resources, change your app's code, or start a deployment.

Where should passwords and API keys go?

Use the app's Environment Variables area when instructed. Do not place real values in code, GitHub files, documentation, screenshots, chat messages, or email.

Can I delete the portal record but keep the live app?

Yes, because deletion scopes are separate. However, if you remove the portal record and keep GitHub or Azure, those resources will no longer appear in My Apps and may require manual IT assistance later.

Can deletion be undone?

Do not assume it can. Deleting a GitHub repository or Azure deployment may permanently remove code, history, configuration, or data. Coordinate and preserve anything required before deleting.

What should I include in a support request?

Include the app name, support reference, approximate time, action selected, and exact on-screen message. Never include passwords, secret values, private keys, or environment-variable contents.

Glossary

App

A website or service created, tracked, and published through the portal.

App owner

The primary person responsible for an app. Owners can manage collaborators and delete scoped resources.

App audience

The people who can use a published app. A new app can require Cedarville sign-in or be openly public on the internet.

Azure

Microsoft's cloud platform. The portal uses Azure to host running apps and related app-specific resources.

Codex

An AI coding assistant that can help create or change an app's files. Finished changes must reach the managed GitHub repository before the portal can publish them.

Collaborator

A Cedarville coworker invited to help manage an app in the portal. GitHub repository access is requested separately.

Database

Structured storage used when an app must remember records such as requests, approvals, or tracker entries.

Deployment

A particular attempt to build and send the app to Azure. Publishing starts a deployment.

Environment variable

A named runtime setting supplied outside the app's code. It may contain a normal configuration value or a secret.

Git

The software that records a history of changes in the app folder on your computer. For portal work, install it from Company Portal on Windows or CedarNet 2.0 on macOS. Codex runs the Git commands for you.

GitHub

The service Cedarville uses to store app code, record changes, collaborate, and run publishing workflows.

Local Codex project

A Codex project connected to a folder on your computer. Open or create this project before starting an app task so Codex works with the correct files.

Managed repository

The private GitHub repository created or imported by the portal and treated as the supported source of truth.

Pull request

A GitHub review page that shows proposed file changes before they are merged into the app. The portal uses one instead of overwriting conflicting publishing files.

Microsoft Entra login

Cedarville sign-in used to identify users and control who may enter an app.

Primary folder

The main folder attached to a local Codex project. Codex uses it as the starting location for files and Git work. It should be the new empty app folder for a generated app or the existing app folder for a local app.

Publish

Build and send the current managed GitHub version to Azure so it can run in a web browser.

Share in Portal

An optional directory listing visible to people signed in to CU Launch. It helps them discover an app but does not change the app audience.

Publishing setup

The GitHub workflow, Azure resources, secrets, and credentials that allow the portal to publish an app safely.

Quick chat

A conversation that is not attached to the app's local Codex project. Do not use Quick chat for a portal Git handoff; open the local project and start the task inside it.

Source of truth

The managed GitHub copy that Cedarville tools treat as the current app. Local changes must be uploaded there before the portal can publish them.

Support reference

A safe identifier that helps Cedarville IT find technical records for a failed or important action.

Template

An approved starting point that supplies common app files, features, and Cedarville defaults.