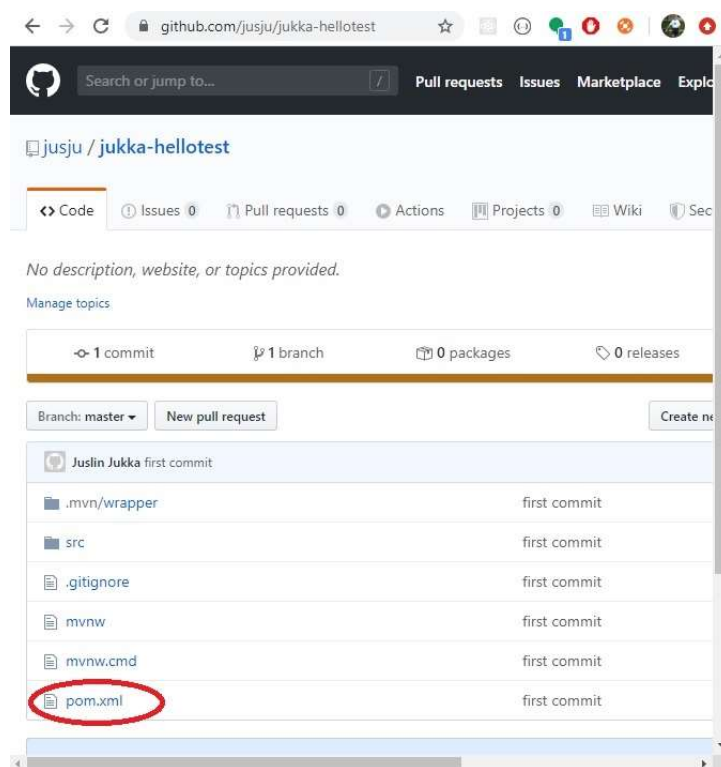


Heroku Deployment Instructions with PostgreSQL

1. Create a Spring Boot project in your Eclipse or use as a test in course demo - for example **"SecureStudentListUser"** should do, because it has at least both the database and Spring Security, which is a minimum software should have. That software needs to be modified though from **application.properties** to match needs of a real **Postgre** database.
2. Make sure your project is in [github](#) from its root level, meaning that when you open a given repository you see for example pom.xml on the root level:

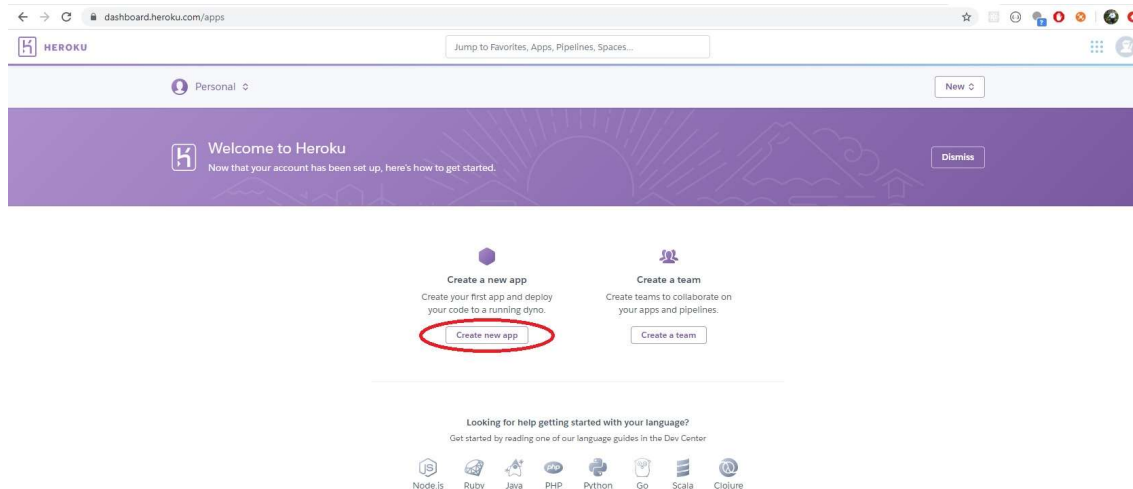


You can put your project to github using git bash program, if you go with cd (change directory) command to the root level of a specific project. Then you would issue following commands, and after these commands, you would find the project in github.

```
git init
git add .
git commit -m "first commit"
git remote add origin https://github.com/USERNAME/REPONAME.git
git push
```

3. Open Heroku and get an account in Heroku if you do not have one.
(<https://www.heroku.com/>)

After signed in you should see a screen something like the following. Commercial services change their layout all the time so this picture will not be updated. Essential is to find a button, that says something like “Create a new app”.



4. Create an app in Heroku (<https://dashboard.heroku.com/new-app>)

After clicked on “Create new app”, then you find the following dialog. Simple fill the form by giving a name to your app, and choosing a region (US or Europe). Press the button “Create app” and continue.

A screenshot of the 'Create New App' dialog in Heroku. The dialog has a light blue header with the Heroku logo and a search bar. The main content area has a form with two fields: 'App name' and 'Choose a region'. The 'App name' field has the placeholder text 'app-name'. The 'Choose a region' field has a dropdown menu with 'Europe' selected. Below these fields, there's a link 'Add to pipeline...' and a 'Create app' button.

Please note that, you would get tips of naming - lowercase letters, dash, digits are allowed. Try and get a satisfied name for your app.

5. Deployment methods in heroku.

After you've created an app in heroku, you will be guided to deployment page. Heroku has their own git system, but the most direct way is probably to connect to your git repository on Github by clicking on **"GitHub - connect to Github"**.

The screenshot shows the Heroku deployment page for an app named 'canalysis'. The top navigation bar includes 'Personal', 'canalysis', and buttons for 'Open app' and 'More'. Below the navigation bar are tabs for 'Overview', 'Resources', 'Deploy', 'Metrics', 'Activity', 'Access', and 'Settings'. The main content area is divided into two sections: 'Add this app to a pipeline' and 'Add this app to a stage in a pipeline to enable additional features'. The 'Add this app to a pipeline' section includes a dropdown menu to 'Choose a pipeline'. The 'Add this app to a stage in a pipeline' section includes a 'Choose a pipeline' dropdown and a 'Learn more' link. Below these sections is a 'Deployment method' section with three options: 'Heroku Git Use Heroku CLI', 'GitHub Connect to GitHub' (highlighted with a blue circle), and 'Container Registry Use Heroku CLI'. Below the 'Deployment method' section is a 'Deploy using Heroku Git' section with instructions on how to use git in the command line or a GUI tool to deploy this app. Below the 'Deploy using Heroku Git' section is an 'Install the Heroku CLI' section with instructions on how to download and install the Heroku CLI, and a 'heroku login' button.

6. Connect to GitHub repository

Once you are connected to GitHub, a dialog of search for a repository from your Github repositories. Click on "Search", all the available repositories are displayed, and press "Connect" for the one you'd like to deploy on heroku.



The screenshot shows the Heroku deployment page for an app named 'canalysis'. The top navigation bar includes 'Personal', 'canalysis', and buttons for 'Open app' and 'More'. Below the navigation bar are tabs for 'Overview', 'Resources', 'Deploy', 'Metrics', 'Activity', 'Access', and 'Settings'. The main content area is divided into two sections: 'Add this app to a pipeline' and 'Add this app to a stage in a pipeline to enable additional features'. The 'Add this app to a pipeline' section includes a dropdown menu to 'Choose a pipeline'. The 'Add this app to a stage in a pipeline' section includes a 'Choose a pipeline' dropdown and a 'Learn more' link. Below these sections is a 'Deployment method' section with three options: 'Heroku Git Use Heroku CLI', 'GitHub Connect to GitHub' (highlighted with a yellow circle), and 'Container Registry Use Heroku CLI'. Below the 'Deployment method' section is a 'Connect to GitHub' section with instructions on how to connect this app to GitHub to enable code diffs and deploys. Below the 'Connect to GitHub' section is a 'Search for a repository to connect to' section with a search bar and a 'Search' button. Below the search bar is a list of repositories with their names and a 'Connect' button for each. The repositories listed are: '/WebGL-Earth-API', '/RandomUserDisplay', '/ReactGql', '/CustomerList_JDBC', '/Bookstore', and '/Todolist_materialUI'.

7. App deployment

As you can see your Github repository is connected to the app, now you have two options:

a). "Enable automatic deploys from Github"; b). "Manual deploy a Github branch".

App connected to GitHub
Code diffs, manual and auto deploys are available for this app.

Connected to  `"/Todolist_materialUI"` by  `heroku`
[Disconnect...](#)

Releases in the [activity feed](#) link to GitHub to view commit diffs

Automatic deploys
Enables a chosen branch to be automatically deployed to this app.

Enable automatic deploys from GitHub
Every push to the branch you specify here will deploy a new version of this app. **Deploys happen automatically:** be sure that this branch is always in a deployable state and any tests have passed before you push. [Learn more.](#)

Choose a branch to deploy

master

☐ Wait for CI to pass before deploy
Only enable this option if you have a Continuous Integration service configured on your repo.

Enable Automatic Deploys

Manual deploy
Deploy the current state of a branch to this app.

Deploy a GitHub branch
This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

master

Deploy Branch

Note that, the automatic deployment will update your app on heroku for every push you made to github, say it the newest version. If this is the case, you'd press "Enable Automatic Deploys".

Otherwise, go "Manual deploy", to "Deploy a Github branch", so that you get the build dialog after pressing "Deploy Branch". The latest git commit is shown in "Build master".

Manual deploy
Deploy the current state of a branch to this app.

Deploy a GitHub branch
This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

master

Deploy Branch

Receive code from GitHub

Build master bcd8a6fe9

```
flowtype@3.x".
warning "react-scripts > @typescript-eslint/eslint-plugin > tsutils@3.17.1" has unmet peer dependency
"typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta
|| >= 3.7.0-dev || >= 3.7.0-beta".
[4/4] Building fresh packages...
warning Ignored scripts due to flag.
Done in 11.73s.

-----> Caching build
Autoscroll with output
```

[View build log](#)

Release phase

Deploy to Heroku

8. Deployment succeeds

In case everything went well, you would get all greens for build and release. Finally a url is provided to run the deployed app. Click on “View”.

Manual deploy

Deploy the current state of a branch to this app.

Deploy a GitHub branch

This will deploy the current state of the branch you specify below. [Learn more.](#)

Choose a branch to deploy

master

Deploy Branch

Receive code from GitHub

✓

Build master bc8a6fe9

✓

Release phase

✓

Deploy to Heroku

✓

Your app was successfully deployed.

[View](#)

9. Activity log and Build log

Not every time we get successful stories. In case of failure, It is quite useful to check on Activity log and Build log. The activity log is under the main menu of personal app. Click on “view build log”, you see details of compiling, building, packaging and exceptions.

Overview

Resources

Deploy

Metrics

Activity

Access

Settings

Activity Feed

@gmail.com: Deployed bc8a6fe9

Today at 8:24 PM · v3

@gmail.com: Build succeeded

Today at 8:21 PM · [View build log](#)

@gmail.com: Build failed

Today at 8:19 PM · [View build log](#)

@gmail.com: Build failed

Today at 8:09 PM · [View build log](#)

@gmail.com: Enable Logplex

Today at 6:37 PM · v2 · [Roll back to here](#)

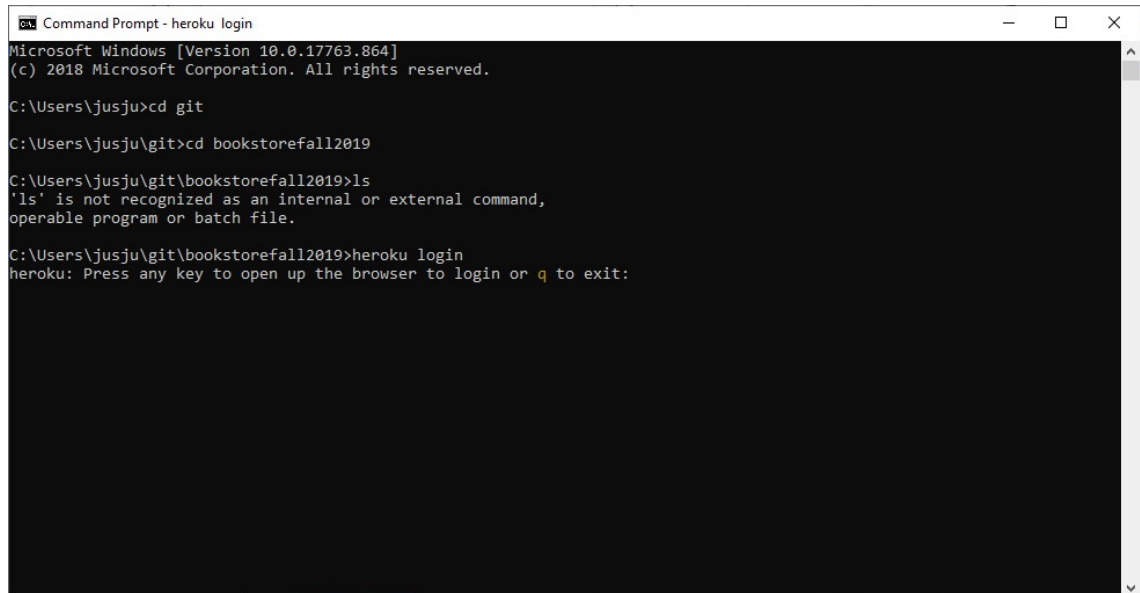
@gmail.com: Initial release

Today at 6:37 PM · v1 · [Roll back to here](#)

- During Heroku usage you probably need to use heroku cli. Install heroku cli and use it from Windows command prompt, not git bash. cli means command line interface

- First do command:

```
heroku login
```



```
Command Prompt - heroku login
Microsoft Windows [Version 10.0.17763.864]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\jusju>cd git
C:\Users\jusju\git>cd bookstorefall2019
C:\Users\jusju\git\bookstorefall2019>ls
'ls' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\jusju\git\bookstorefall2019>heroku login
heroku: Press any key to open up the browser to login or q to exit:
```

- Login to the system and after that you can see for example logs with this command. Test it, though you probably will not see anything on first use. You might have to repeat command two times. The point is to test and see if you can see error.

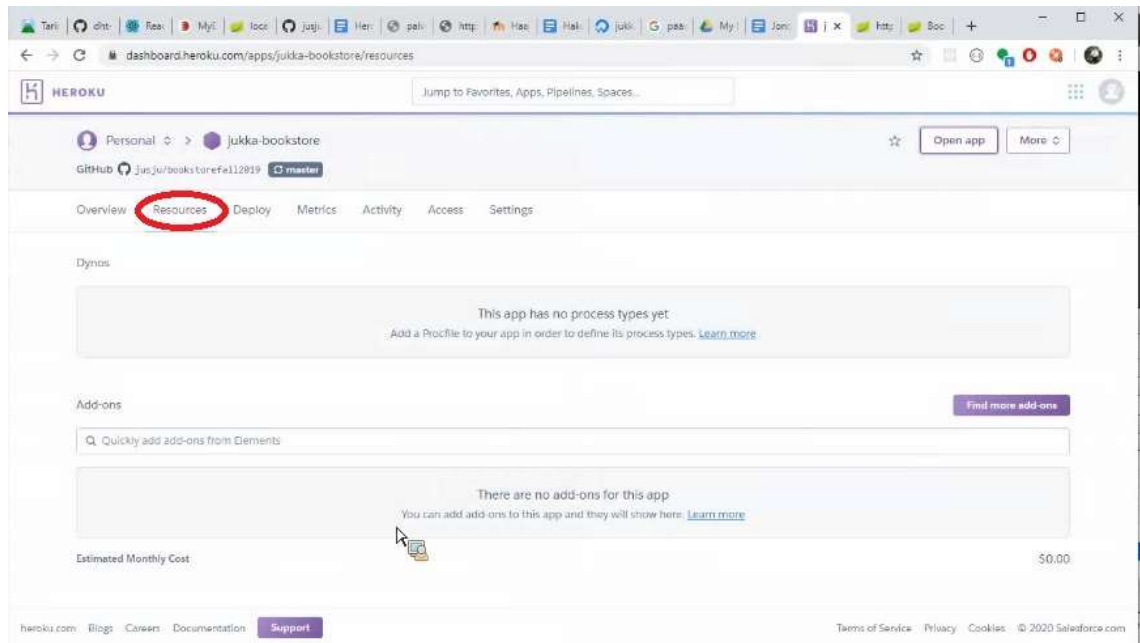
```
heroku logs --app jukka-bookstore --tail
```

Replace jukka-bookstore with your application name.

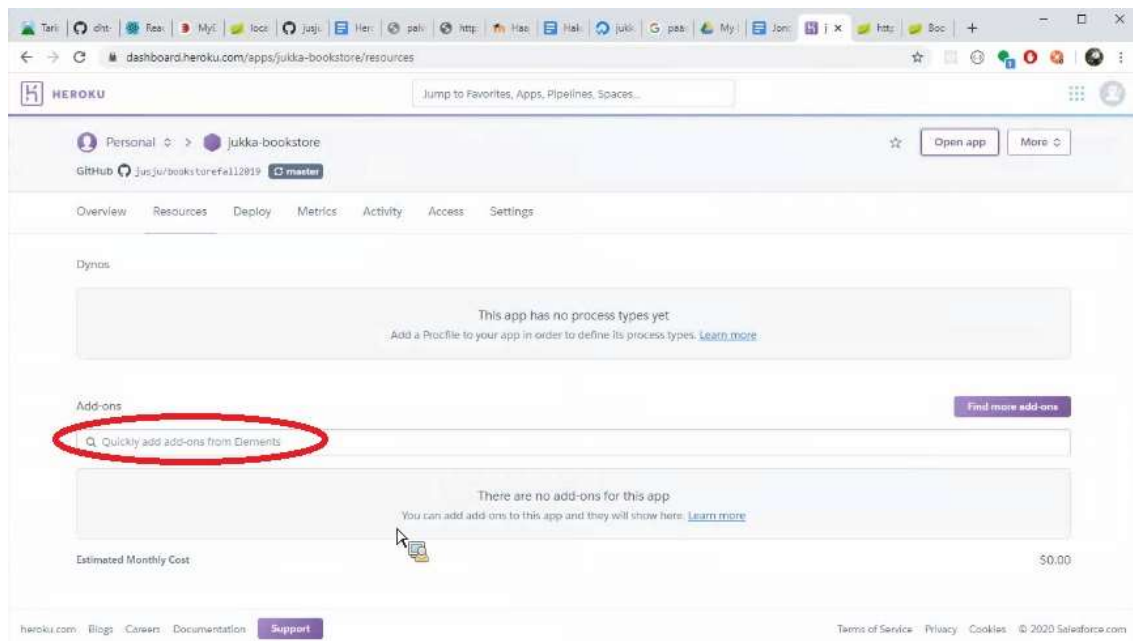
- Add to your **pom.xml**:

```
<dependency>
  <groupId>org.postgresql</groupId>
  <artifactId>postgresql</artifactId>
</dependency>
```

- Remove H2 dependency from your pom.xml
- Go to **Resources** in your heroku application view.

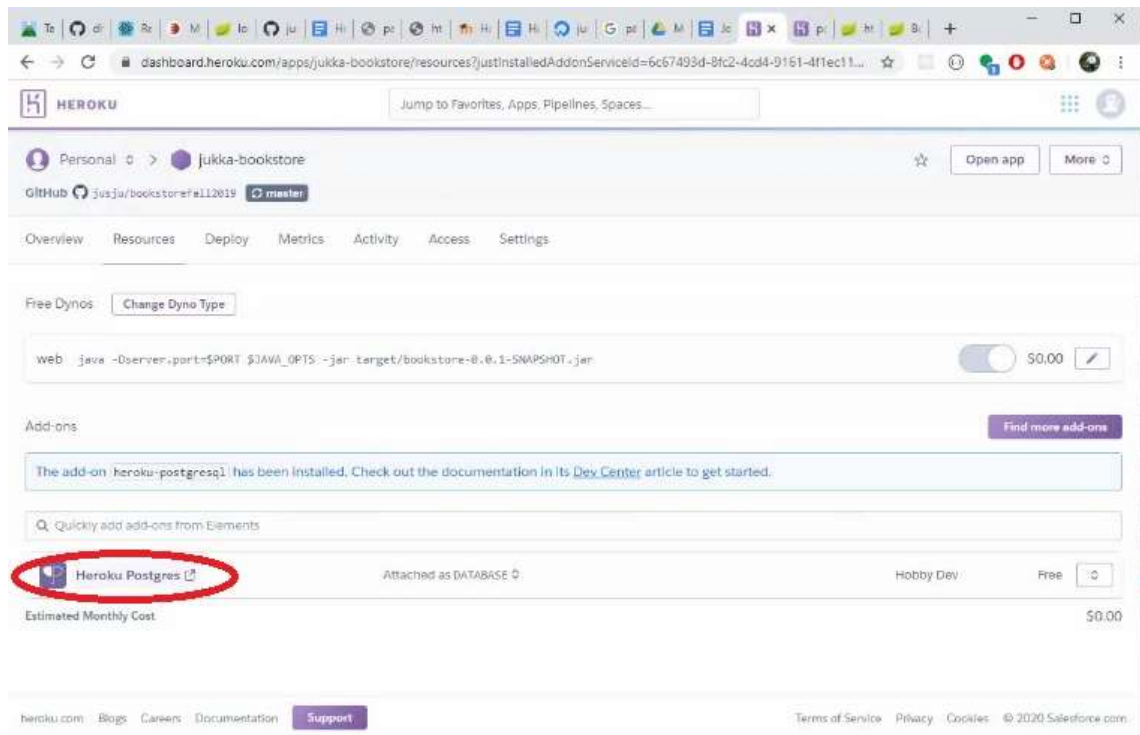


- Type to **Add-Ons** field:
Heroku Postgres

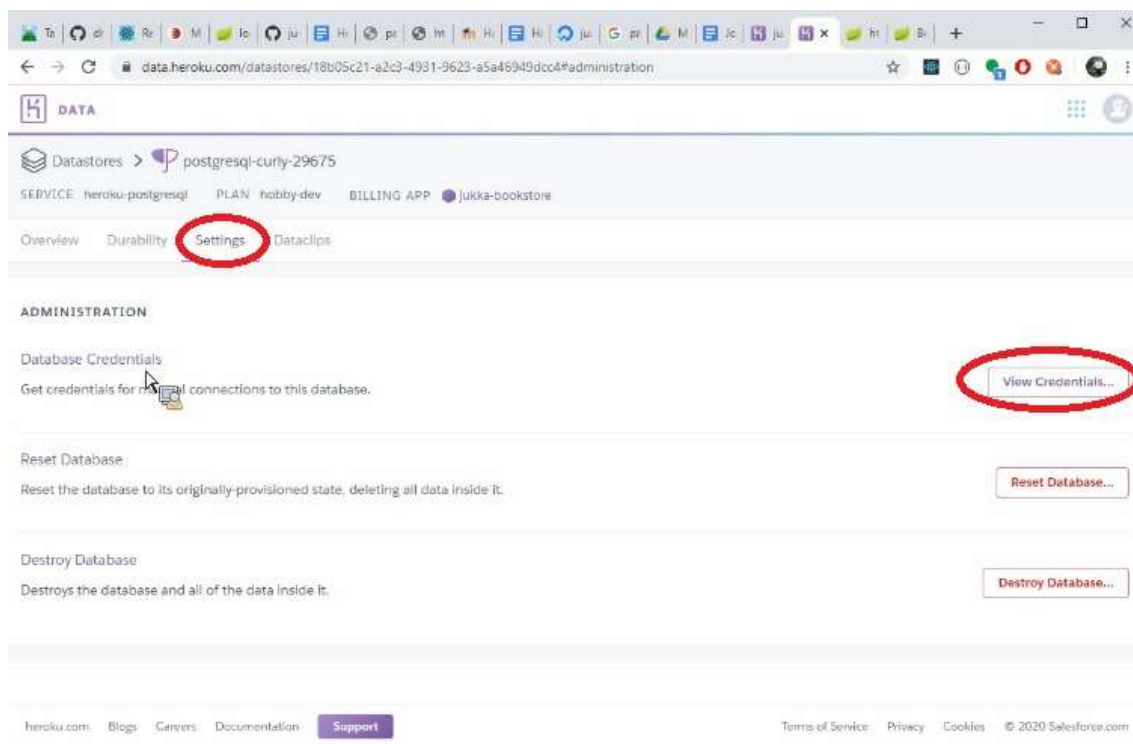


- Select Free plan.

- Click on **Heroku Postgres**.



- Open go to **Settings -> View Credentials**.



- Copy credentials from that screen to your `application.properties` to format given here. Change credentials to be your credentials.

```
#spring.h2.console.enabled=true
#spring.h2.console.path=/h2-console
#spring.datasource.url=jdbc:h2:mem:testdb
logging.level.org.hibernate.type.descriptor.sql.BasicBinder=trace
spring.data.rest.basePath=/api

spring.jpa.show-sql=true
#spring.datasource.url=jdbc:h2:mem:testdb
#spring.h2.console.enabled=true
#spring.h2.console.path=/h2-console
spring.jpa.open-in-view=true
spring.datasource.driver-class-name=org.postgresql.Driver

server.port=8080
spring.jpa.database=POSTGRESQL
spring.datasource.platform=postgres
#spring.datasource.url=jdbc:postgresql://ec2-46-137-156-205.eu-west-1.compute.amazonaws.com:5432/dipili864du5fp?sslmode=require
#spring.datasource.url=jdbc:postgresql://ec2-54-247-103-43.eu-west-1.compute.amazonaws.com:5432/d6ah4utt1744t5?sslmode=require
# possible values for hibernate.ddl-auto are create-drop, validate, none and update

#spring.datasource.username=CHANGE_ME
#spring.datasource.password=CHANGE_ME

spring.datasource.url=jdbc:postgresql://ec2-54-74-35-87.eu-west-1.compute.amazonaws.com:5432/d2kconh7cucm25?sslmode=require
spring.datasource.username=CHANGE_ME
spring.datasource.password=CHANGE_ME

spring.jpa.show-sql=true
#spring.jpa.generate-ddl=true
spring.jpa.hibernate.ddl-auto=update
spring.datasource.initialization-mode=always

spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true

#spring.datasource.url=${SPRING_DATASOURCE_URL}
#spring.datasource.username=${SPRING_DATASOURCE_USERNAME}
#spring.datasource.password=${SPRING_DATASOURCE_PASSWORD}
```

Note # hashtag is mark for comment.

Username are UNIQUE, so you must deleteAll in Heroku.

Default success url better to have two mappings at controller.

If you have **User** named entity, it conflicts with PostgreSQL user reserved work so **rename your user to something else. For example fix like this:**

```
@Entity
@Table(name="usertable")
public class User {
```

For security reasons try not to publish to github your credentials:

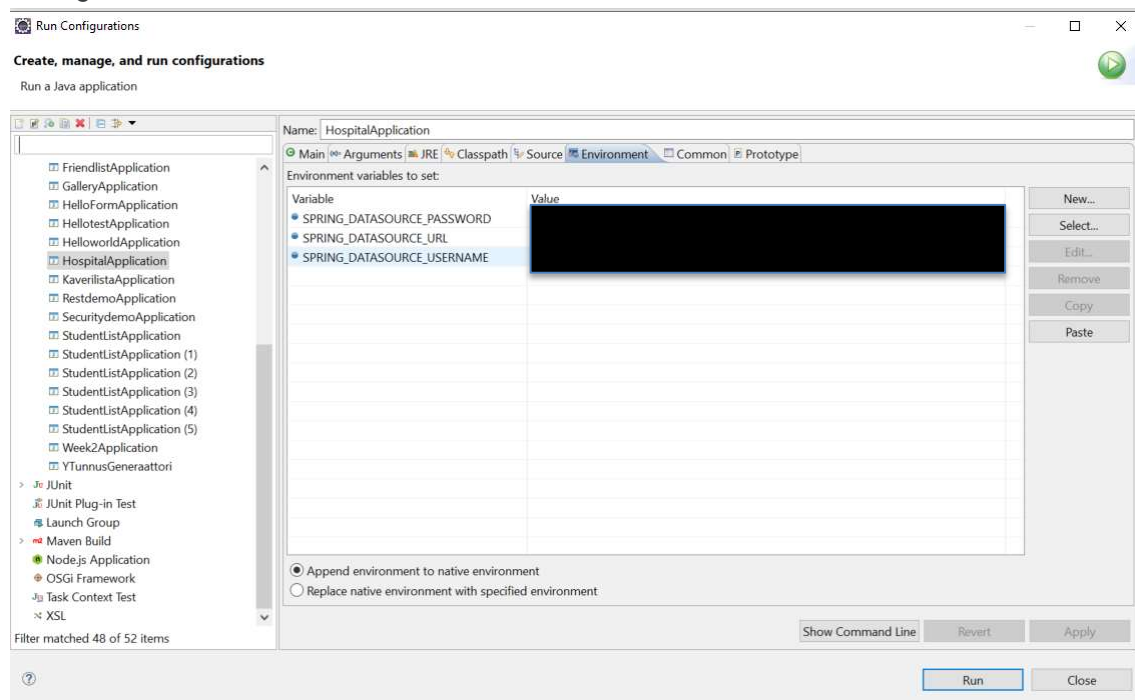
Once the database add-on has been created, Heroku will automatically populate the environment variables

```
spring.datasource.url=${SPRING_DATASOURCE_URL}
```

```
spring.datasource.username=${SPRING_DATASOURCE_USERNAME}
```

```
spring.datasource.password=${SPRING_DATASOURCE_PASSWORD}
```

If you still want the software also to work on local environment add to Run As... configurations env variables like this:



Remember also in command line runner to delete previous:

urepo

Heroku has several limitations:

- by default has a long sleep mode to start an application
- cannot send email etc so cannot implement forgot your password feature in a normal way
- can only use postgres for free, not for example mariadb
- tries only to start app for 30 second, if does not succeed in that time quits
- supports java 1.8