



# PYTHON BOOTCAMP

[www.jomhack.com](http://www.jomhack.com)

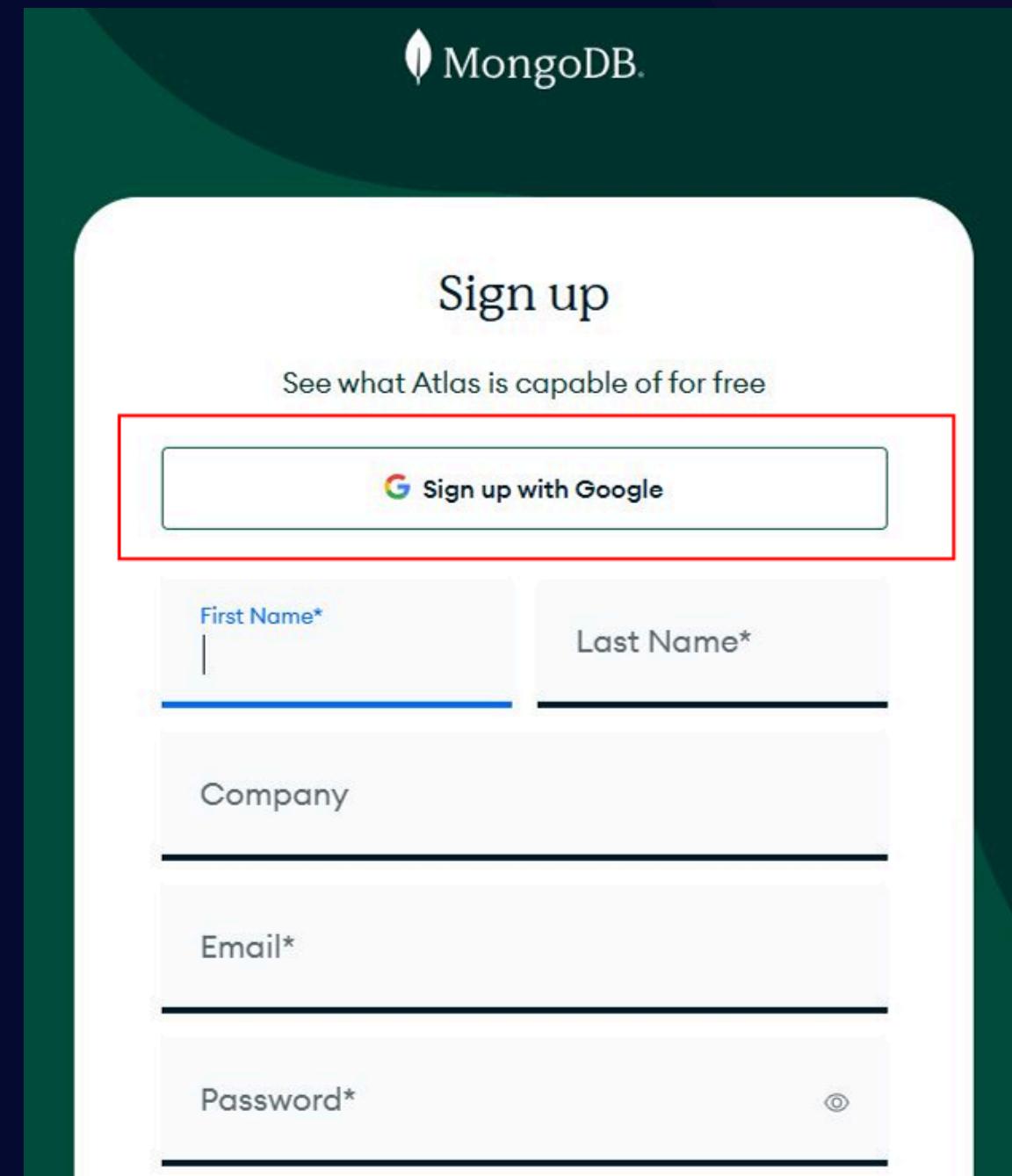
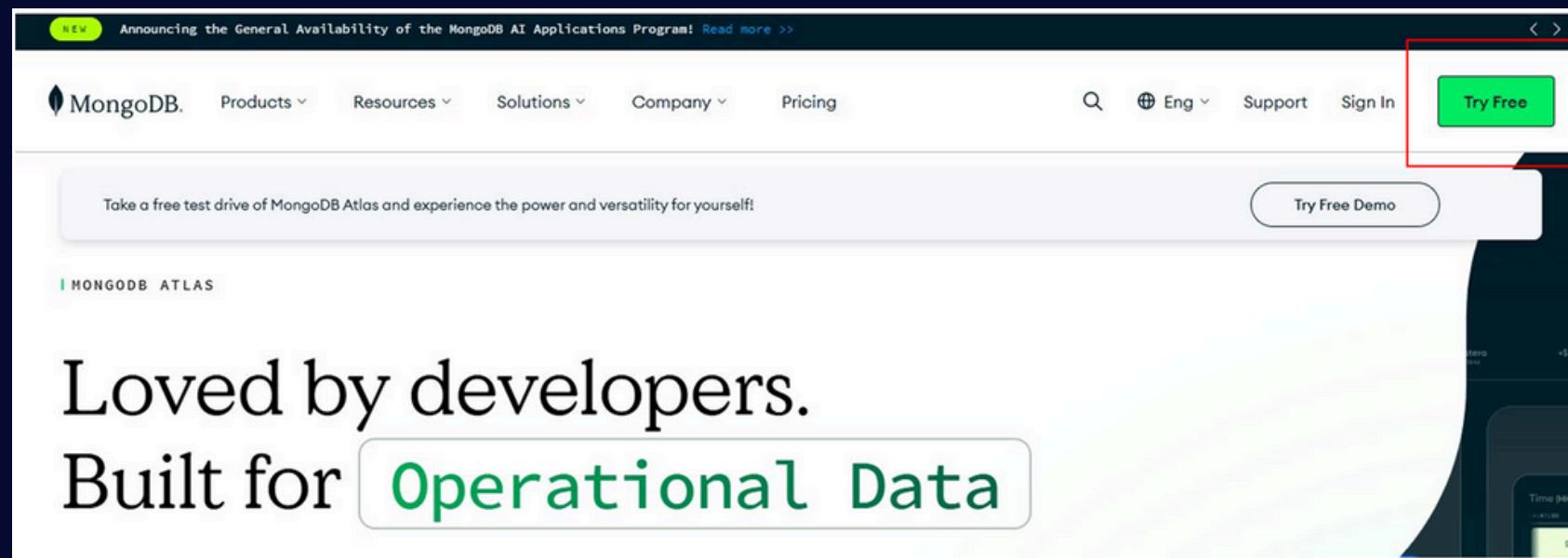
# DATABASE (MONGO)

## MongoDB Atlas:

- <https://www.mongodb.com>
- Cloud database service for MongoDB
- 512mb free tier
- pip install pymongo

# DATABASE (MONGO)

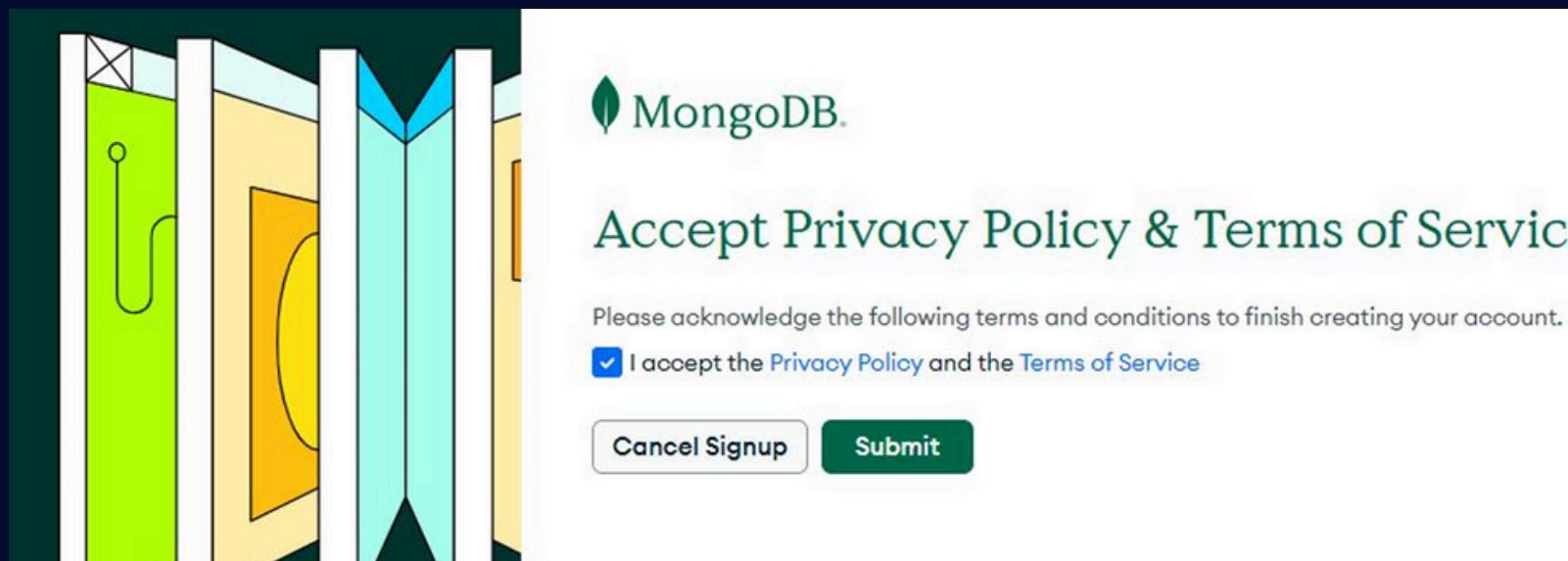
## MongoDB Setup:



The screenshot shows the MongoDB sign-up page. The title 'MongoDB.' is at the top. Below it, a large button labeled 'Sign up' is followed by the text 'See what Atlas is capable of for free'. A 'Sign up with Google' button is highlighted with a red box. The form fields include 'First Name\*', 'Last Name\*', 'Company', 'Email\*', and 'Password\*'. There is also a 'Forgot Password?' link.

# DATABASE (MONGO)

## MongoDB Setup:



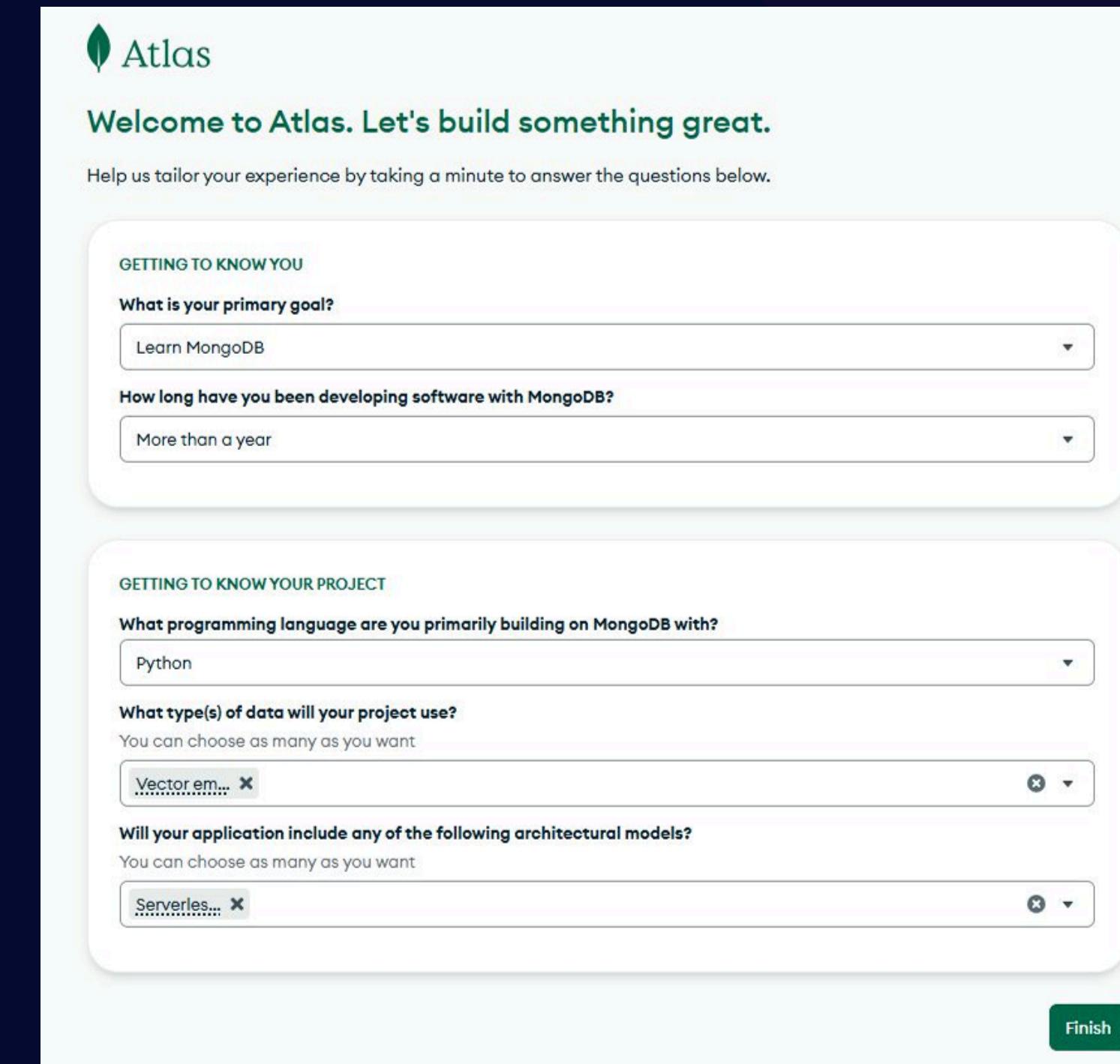
MongoDB.

### Accept Privacy Policy & Terms of Service

Please acknowledge the following terms and conditions to finish creating your account.

I accept the [Privacy Policy](#) and the [Terms of Service](#)

[Cancel Signup](#) [Submit](#)



Atlas

### Welcome to Atlas. Let's build something great.

Help us tailor your experience by taking a minute to answer the questions below.

**GETTING TO KNOW YOU**

**What is your primary goal?**  
Learn MongoDB

**How long have you been developing software with MongoDB?**  
More than a year

**GETTING TO KNOW YOUR PROJECT**

**What programming language are you primarily building on MongoDB with?**  
Python

**What type(s) of data will your project use?**  
You can choose as many as you want  
Vector em... X

**Will your application include any of the following architectural models?**  
You can choose as many as you want  
Serverless... X

[Finish](#)

# DATABASE (MONGO)

## MongoDB Setup:

**Deploy your cluster**

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

<input type="radio"/> M10      \$0.10/hour	<input type="radio"/> Serverless      \$0.12/1M reads	<input checked="" type="radio"/> M0      Free
For learning and exploring MongoDB in a cloud environment.		
STORAGE 10 GB	RAM 2 GB	vCPU 2 vCPUs
Up to 1TB	Auto-scale	Auto-scale
512 MB      Shared      Shared		

**Free forever!** Your free cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

**Name**  
You cannot change the name once the cluster is created.

Automate security setup i

Preload sample dataset i

**Provider**  
 aws  Google Cloud  Azure

**Region**  
 ★ Recommended i  Low carbon emissions i

**Choose Google Cloud**

**Deploy the cluster**

**Connect to Cluster0**

- Set up connection security
- Choose a connection method
- Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

**1. Add a connection IP address**

Your current IP address (140.82.192.204) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).

**2. Create a database user**

This first user will have [atlasAdmin](#) permissions for this project.

We autogenerated a username and password. You can use this or create your own.

i You'll need your database user's credentials in the next step. Copy the database user password.

<b>Username</b> <input type="text" value="teobeequan"/>	<b>Password</b> <input type="text" value="8k6brDNPR7GgiZ2c"/> <small>HIDE</small> <input type="button" value="Copy"/>
--	--

**Create Database User**

# DATABASE (MONGO)

## MongoDB Setup:

**Connect to Cluster0**

- 1 Set up connection security
- 2 Choose a connection method
- 3 Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

**1. Add a connection IP address**

Your current IP address (140.82.192.204) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).

**2. Create a database user**

A database user has been added to this project. Create another user later in [Database Access](#). You'll need your database user's credentials in the next step.

**Choose a connection method**

**Close**

**Connect to Cluster0**

- 1 Set up connection security
- 2 Choose a connection method
- 3 Connect

**Connect to your application**

-  **Drivers**  
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

**Access your data through tools**

-  **Compass**  
Explore, modify, and visualize your data with MongoDB's GUI
-  **Shell**  
Quickly add & update data using MongoDB's Javascript command-line interface
-  **MongoDB for VS Code**  
Work with your data in MongoDB directly from your VS Code environment
-  **Atlas SQL**  
Easily connect SQL tools to Atlas for data analysis and visualization

**Go Back** **Close**

# DATABASE (MONGO)

## MongoDB Setup:

Set up connection security   Choose a connection method   Connect

Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.

Driver	Version
Python	3.12 or later

2. Install your driver

Run the following on the command line  
Note: Use appropriate Python 3 executable

```
python -m pip install "pymongo[srv]"
```

[View MongoDB Python Driver installation instructions.](#)

3. Add your connection string into your application code

Use this connection string in your application

View full code sample

```
mongodb+srv://<REDACTED>@cluster0.osrra.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0
```

Replace `<db_password>` with the password for the `teobguan2013` database user. Ensure any option params are URL encoded.

RESOURCES

- [Get started with the Python Driver](#)
- [Access your Database Users](#)
- [Python Starter Sample App](#)
- [Troubleshoot Connections](#)

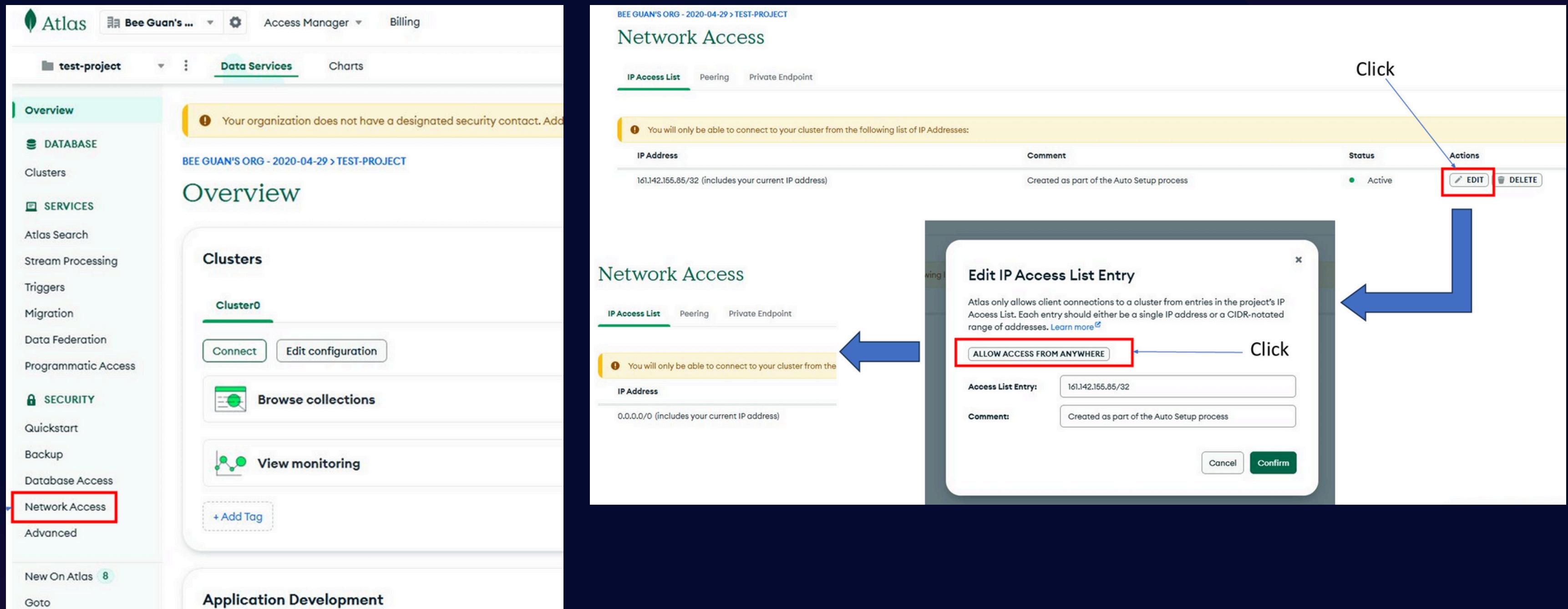
.env

Import to Postman

1 MONGODB\_ATLAS\_CLUSTER\_URI=mongodb+srv://  
2

# DATABASE (MONGO)

## MongoDB Setup:



The image shows a step-by-step guide for configuring MongoDB Atlas Network Access. It consists of three panels: a left navigation panel, a middle configuration panel, and a right modal dialog.

**Left Panel (Atlas Overview):**

- Project: test-project
- Section: Data Services (highlighted)
- Sub-section: Overview
- Clusters: Cluster0 (highlighted)
- Actions: Connect, Edit configuration
- Services: Browse collections, View monitoring
- Security: Quickstart, Backup, Database Access, Network Access (highlighted), Advanced
- Bottom: New On Atlas 8, Goto

**Middle Panel (Network Access):**

BEE GUAN'S ORG - 2020-04-29 > TEST-PROJECT

### Network Access

IP Access List (highlighted)

IP Address	Comment	Status	Actions
161.142.155.85/32 (includes your current IP address)	Created as part of the Auto Setup process	Active	<span>EDIT</span> <span>DELETE</span>

**Right Panel (Edit IP Access List Entry):**

Click → **Edit IP Access List Entry**

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more](#)

**ALLOW ACCESS FROM ANYWHERE** (highlighted) → Click

Access List Entry: 161.142.155.85/32

Comment: Created as part of the Auto Setup process

Cancel Confirm

# DATABASE (MONGO)

## MongoDB Setup:

Cluster0

Monitoring for Cluster0 is Paused  
Monitoring will automatically resume when you connect to your cluster.  
[Visit the documentation](#) for more info.

VERSION	REGION	TYPE	BACKUPS	LINKED APP SERVICES	ATLAS SQL	ATLAS SEARCH
8.0.13	AWS / Singapore (ap-southeast-1)	Replica Set - 3 nodes	Inactive	None Linked	Connect	3 search indexes

+ Add Tag

Overview Real Time Metrics Collections Atlas Search Query Insights Performance Advisor Online

DATABASES: 3 COLLECTIONS: 10

+ Create Database

Search Namespaces

example\_db

- posts
  - users
  - sample\_mflix
  - test\_db

example\_db.posts

STORAGE SIZE: 4KB LOGICAL DATA SIZE: 0B TOTAL DOCUMENTS: 0 INDEXES TOTAL SIZE: 8KB

Find Indexes Schema Anti-Patterns 0 Aggregation Search Indexes

Generate queries from natural language in Compass

INSERT DOCUMENT

Filter Type a query: { field: 'value' } Reset Apply Options ▾

QUERY RESULTS: 0

# DATABASE (MONGO)

## Initialize:

- pip install pymongo
- pip install python-dotenv

```
2  from pymongo import MongoClient
3  from datetime import datetime
4  from bson.objectid import ObjectId
5  from dotenv import load_dotenv
6  import os
7
8  load_dotenv() # Load environment variables from .env file
9
10 mongo_uri = os.getenv('MONGODB_ATLAS_CLUSTER_URI')
11
12 class DatabaseManager:
13     def __init__(self, db_name='example_db', connection_string=mongo_uri):
14         self.client = MongoClient(connection_string)
15         self.db = self.client[db_name]
16         self.users_collection = self.db.users
17         self.posts_collection = self.db.posts
18         self.init_database()
19
20     def init_database(self):
21         """Initialize database with collections and indexes"""
22         # Create unique index on email for users
23         self.users_collection.create_index("email", unique=True)
24         # Create index on user_id for posts for better query performance
25         self.posts_collection.create_index("user_id")
```

# DATABASE (MONGO)

## Create Function:

```

27     def create_user(self, name, email, age):
28         """Create a new user"""
29         try:
30             user_doc = {
31                 "name": name,
32                 "email": email,
33                 "age": age,
34                 "created_at": datetime.now()
35             }
36             result = self.users_collection.insert_one(user_doc)
37             return str(result.inserted_id)
38         except Exception as e:
39             print(f"Error: {e}")
40             return None

```

## Create Function:

```

42     def create_post(self, user_id, title, content):
43         """Create a new post"""
44         try:
45             # Convert string user_id to ObjectId if it's a valid ObjectId
46             if ObjectId.is_valid(user_id):
47                 user_object_id = ObjectId(user_id)
48             else:
49                 user_object_id = user_id
50
51             post_doc = {
52                 "user_id": user_object_id,
53                 "title": title,
54                 "content": content,
55                 "created_at": datetime.now()
56             }
57             result = self.posts_collection.insert_one(post_doc)
58             return str(result.inserted_id)
59         except Exception as e:
60             print(f"Error creating post: {e}")
61             return None

```

# DATABASE (MONGO)



## Read Function:

```
63     def get_all_users(self):
64         """Get all users"""
65         try:
66             users = list(self.users_collection.find())
67             # Convert ObjectId to string for display
68             for user in users:
69                 user['_id'] = str(user['_id'])
70             return users
71         except Exception as e:
72             print(f"Error fetching users: {e}")
73             return []
```

## Read Function:

```
75     def get_user_posts(self, user_id):
76         """Get posts by user"""
77         try:
78             # Convert string user_id to ObjectId if it's a valid ObjectId
79             if ObjectId.is_valid(user_id):
80                 user_object_id = ObjectId(user_id)
81             else:
82                 user_object_id = user_id
83
84             posts = list(self.posts_collection.find(
85                 {"user_id": user_object_id}
86             ).sort("created_at", -1))
87
88             # Convert ObjectId to string for display
89             for post in posts:
90                 post['_id'] = str(post['_id'])
91                 post['user_id'] = str(post['user_id'])
92
93             return posts
94         except Exception as e:
95             print(f"Error fetching posts: {e}")
96             return []
```

# DATABASE (MONGO)

## Delete Function:

```

98     def delete_user(self, user_id):
99         """Delete user and their posts"""
100        try:
101            # Convert string user_id to ObjectId if it's a valid ObjectId
102            if ObjectId.is_valid(user_id):
103                user_object_id = ObjectId(user_id)
104            else:
105                user_object_id = user_id
106
107            # Delete user's posts first
108            self.posts_collection.delete_many({"user_id": user_object_id})
109
110            # Delete the user
111            result = self.users_collection.delete_one({"_id": user_object_id})
112            return result.deleted_count > 0
113        except Exception as e:
114            print(f"Error deleting user: {e}")
115        return False

```

## Close DB Function:

```

117    def close_connection(self):
118        """Close the MongoDB connection"""
119        self.client.close()

```

# DATABASE (MONGO)

## Run on Terminal Function:

```

121  def display_menu():
122      """Display the main menu"""
123      print("\n" + "="*40)
124      print("          DATABASE MANAGER")
125      print("="*40)
126      print("1. Create User")
127      print("2. View All Users")
128      print("3. Create Post")
129      print("4. View User Posts")
130      print("5. Delete User")
131      print("6. Exit")
132      print("-"*40)

```

```

134  def main():
135      """Main interactive CLI function"""
136      try:
137          db = DatabaseManager()
138          print("/ Connected to MongoDB successfully!")
139      except Exception as e:
140          print(f"X Failed to connect to MongoDB: {e}")
141          print("Make sure MongoDB is running on localhost:27017")
142

```

# DATABASE (MONGO)

## Run on Terminal Function:

```
144     while True:
145         display_menu()
146         choice = input("Enter your choice (1-6): ").strip()
147
148         if choice == '1':
149             print("\n--- Create New User ---")
150             name = input("Enter name: ").strip()
151             email = input("Enter email: ").strip()
152             try:
153                 age = int(input("Enter age: ").strip())
154                 user_id = db.create_user(name, email, age)
155                 if user_id:
156                     print(f"✓ User created successfully! ID: {user_id}")
157                 else:
158                     print("✗ Failed to create user")
159             except ValueError:
160                 print("✗ Invalid age. Please enter a number.")
161
162         elif choice == '2':
163             print("\n--- All Users ---")
164             users = db.get_all_users()
165             if users:
166                 for user in users:
167                     print(f"ID: {user['_id']} | Name: {user['name']} | Email: {user['email']} | Age: {user['age']}")
168             else:
169                 print("No users found.")
```

# DATABASE (MONGO)



## Run on Terminal Function:

```
171     elif choice == '3':
172         print("\n--- Create New Post ---")
173         user_id = input("Enter user ID: ").strip()
174         title = input("Enter post title: ").strip()
175         content = input("Enter post content: ").strip()
176         post_id = db.create_post(user_id, title, content)
177         if post_id:
178             print(f"✓ Post created successfully! ID: {post_id}")
179         else:
180             print("✗ Failed to create post")
181
182     elif choice == '4':
183         print("\n--- View User Posts ---")
184         user_id = input("Enter user ID: ").strip()
185         posts = db.get_user_posts(user_id)
186         if posts:
187             for post in posts:
188                 print(f"\nPost ID: {post['_id']}")
189                 print(f"Title: {post['title']}")
190                 print(f"Content: {post['content']}")
191                 print(f"Created: {post['created_at']}")
192                 print("-" * 30)
193         else:
194             print("No posts found for this user.")
```

```
196     elif choice == '5':
197         print("\n--- Delete User ---")
198         user_id = input("Enter user ID to delete: ").strip()
199         confirm = input(f"Are you sure you want to delete user {user_id}? (y/N): ").strip().lower()
200         if confirm == 'y':
201             if db.delete_user(user_id):
202                 print("✓ User deleted successfully!")
203             else:
204                 print("✗ User not found or deletion failed.")
205         else:
206             print("Deletion cancelled.")
207
208     elif choice == '6':
209         print("\nClosing database connection...")
210         db.close_connection()
211         print("Goodbye! 🙏")
212         break
213
214     else:
215         print("✗ Invalid choice. Please enter 1-6.")
216
217         input("\nPress Enter to continue...")
218
219     if __name__ == "__main__":
220         main()
```