Data Specification

NoSQL Recap: Since the system uses Firebase (NoSQL database), data is organized as Collections and Documents instead of traditional relational tables. Foreign Key constraints are managed through application logic instead of hard constraints in the DB.

1. Collection: Account
    - Description: Stores user and business account information
    - Sample JSON structure (Format):


```
{
 "userID": "6",
 "email": "example@email.com",
 "password": "hashed_password_string",
 "type": "business"
}
```


    - Property details:

Attribute name	**Data type**	**Constraints**	**Describe**
userID	Number	X	Unique user ID
email	String	Unique, Not Null	Login email
password	String	Hashed	Encrypted password

type	String	"business" or "individual"	Level of login account

2. Collection: BotConfigAgent
    - Description: Save configuration for each bot
    - Sample JSON structure (Format):

```
{
 "botID": "3",
 "typeModel": "gpt",
 "botName": "Customer Support",
 "websiteLink": "https://example.com",
 "uploadFile": "link_to_drive_file",
 "adjustBotResponses": [
      {
        “question”: "what is your company name"
        “ answer”: "the company name is Sun"
      }
      {
        “question”: "What is your boss's name?"
        “answer”: "My boss's name is Khoa"
      }
  ]
}
```

    - Property details:

Attribute name	Data type	Constraints	Describe
botID	Number	X	Bot ID
botName	String	Not Null	Bot Display Name
typeModel	String	Valid values: "gpt 5", "gemini 3 pro", "gemini 3"	Type of AI model used
websiteLink	String	URL	Website link of data learning bot (Scraping)
uploadFile	String	URL (Storage)	Link to file (PDF/Doc) uploaded to Firebase Storage
adjustBotResponses	Array<String>	question & answer	Custom Response Directive (System prompt)

3. Collection: BotAgent
    - Description: Contains embedded bot information and chat history
    - Note: Chat is an array so it will be designed as a Sub-collection in Firebase
    - Sample JSON structure (Format):

```
{
  "botID": "3",
  "embed": "<iframe>code</iframe>",
  "botApi": "api_key_string"
}
// Sub-collection: Chats
"chats": [
    {
        “chatID”: 6_1
        "userID": "user_6",
        "message": "Hello bot",
        “response”: “OK”
        "timestamp": 1765021351
    }
]
```


- Property details:

Attribute name	**Data type**	**Constraints**	**Describe**
botID	Number	X	Bot identifier
embed	String		Mã nhúng bot lên website

botApi	String	Not Null	API Key để kết nối

Sub-collection: chats (Located inside the BotAgent document)

- Description: Save chat history between User and Bot

Attribute name	**Data type**	**Constraints**	**Describe**
chatID	String	X_Y X is for user X Y is for bot X	ID of the chat causing the error - create when unable to answer relationship
userID	Number	X	ID of the person chatting with the Bot
message	String	Not Null	Content of the message sent by the user
response	String	None	Bot reply content
date	Unix Timestamp	Default: Now	Message time

4. Collection: Group

- Description: Manage user groups

- Sample JSON structure (Format):


```
{
 "groupID": "1",
 "groupName": "Marketing Team",
 "ownerID": "6",
 "sharedMemberId": ["2", "3"],
                “sharedBotId” : ["2", "3"]
}
```


- Property details:

Attribute name	**Data type**	**Constraints**	**Describe**
groupID	Number	X	Group identifier
groupName	String	Not Null	Group display name
ownerID	Number	X	ID of the group creator (Group Admin)
sharedMemberID	Array<Number>		List of IDs of members invited to the group
sharedBotID	Array<Number>		List of Bot IDs shared in the group

5. Collection: BillingHistory

- Description: Stores user payment history, invoices and transaction status
- Sample JSON structure (Format):


```
{
 "billID": "4",
 "accountID": "6”,
 "date": "1765021351",
 “relateBotID”: “3”,
 "amount": 200000,
 "status": "paid",
 "invoice": "https://firebasestorage.googleapis.com/.../invoice.pdf"
}
```


- Property details:

Attribute name	**Data type**	**Constraints**	**Describe**
billID	Number	X	Unique invoice identifier
accountID	Number	X	ID of the account making the payment (used to query the user's bill history) - paid relationship
relatedBotID	Number	X	Paid bots - responsible for relationships

date	Unix Timestamp	Not Null	Transaction execution time
amount	Number	Min >= 0	Payment amount
status	String	"paid", "failed"	Transaction status
invoice	String	URL (Storage)	Link to invoice file (PDF/Image) saved on Storage

6. Collection: Ticket
    - Description: Support tickets when the Bot cannot answer
    - Sample JSON structure (Format):

{

"ticketID": "1",

"nameTicket": " Payment error ",

"question": “Why I can not pay? ",

"status": "resolved",

"timeCreated": "1765021351",

"replied": " Refunded ",

“fromChatId”: “6_1”

}

    - Property details:

Attribute name	**Data type**	**Constraints**	**Describe**
ticketID	Number	X	Unique ticket identifier
nameTicket	String	Not Null	Problem Title
status	String	"resolved" or "pending"	Processing status
timeCreated	Unix Timestamp	Default: Now	Time to create ticket
replied	String		Feedback content from admin
chatID	String	X_Y X is for user X Y is for bot X	ID of the chat causing the error - create when unable to answer relationship