Tutorial: Creando un Horario con la API de Tabulación de Tiempo

Este tutorial te guiará a través del proceso de usar la API de Tabulación de Tiempo para crear un horario. Cubriremos cómo configurar la API, hacer solicitudes para crear una instancia de solucionador, agregar restricciones y recuperar el horario en varios formatos.

Requisitos previos

- Un servidor Flask ejecutando la API de Tabulación de Tiempo.
- Una herramienta para hacer solicitudes HTTP, como curl, Postman, o cualquier cliente HTTP en un lenguaje de programación de tu elección.

Configuración de la API

Asegúrate de que tu servidor Flask esté ejecutándose con la API de Tabulación de Tiempo. La API debería estar accesible en `http://localhost:<port>` si la estás ejecutando localmente.

Paso 2: Creando una Instancia de Solucionador

Para crear una instancia de solucionador, necesitas hacer una solicitud POST al endpoint `/solver` con los datos requeridos en formato JSON.

Solicitud

```
POST http://localhost:<port>/solver
Content-Type: application/json

{
  "subjects_name_list":
    ["Programacion","ProgramacionCp","Algebra","AlgebraCP","Analisis","AnalisisCp","
  "dict_subjects_by_time":
    {"Programacion": 1,"ProgramacionCp": 2,"Algebra": 1,"AlgebraCP": 2,"Analisis": 1
  "teachers_names":
    ["Piad","Idania","Celia","Yudivian","DanielL","OmarLogica","CarmentL","ErnestoAr
  "classrooms_names":
    ["1","2","3","4","5","Postgrado"],
  "groups_names":
    ["C111","C112","C113","C114","C115","C311","C312"],
  "dict_group_subject_time": {
    "C111": {
```

```
    "Programacion": 1,
    "ProgramacionCp": 2,
    "Algebra": 1,
    "AlgebraCP": 2,
    "Analisis": 1,
    "AnalisisCp": 2,
    "Logica": 1,
    "LogicaCp": 1

  },
  "C112": {
    "Programacion": 1,
    "ProgramacionCp": 2,
    "Algebra": 1,
    "AlgebraCP": 2,
    "Analisis": 1,
    "AnalisisCp": 2,
    "Logica": 1,
    "LogicaCp": 1

  },
  "C113": {
    "Programacion": 1,
    "ProgramacionCp": 2,
    "Algebra": 1,
    "AlgebraCP": 2,
    "Analisis": 1,
    "AnalisisCp": 2,
    "Logica": 1,
    "LogicaCp": 1

  },
  "C114": {
    "Programacion": 1,
    "ProgramacionCp": 2,
    "Algebra": 1,
    "AlgebraCP": 2,
    "Analisis": 1,
    "AnalisisCp": 2,
    "Logica": 1,
    "LogicaCp": 1

  },
  "C115": {
    "Programacion": 1,
    "ProgramacionCp": 2,
    "Algebra": 1,
    "AlgebraCP": 2,
    "Analisis": 1,
    "AnalisisCp": 2,
    "Logica": 1,
    "LogicaCp": 1
```

```
  },
  "C311": {

    "CompilacionCon": 1,
    "CompilacionCp": 2,
    "RedesCon": 1,
    "RedesCp": 2,
    "OptimizacionCon": 1,
    "OptimizacionCp": 2,
    "AICon": 1,
    "AICp": 2
  },
  "C312": {

    "CompilacionCon": 1,
    "CompilacionCp": 2,
    "RedesCon": 1,
    "RedesCp": 2,
    "OptimizacionCon": 1,
    "OptimizacionCp": 2,
    "AICon": 1,
    "AICp": 2
  }
},
"shifts":
  [1,2,3],
"days":
  [1,2,3,4,5],
"dict_teachers_to_subjects": {
  "Piad": ["Programacion","AICon"],
  "Idania": ["Analisis"],
  "Celia": ["Algebra"],
  "Yudivian": ["Logica"],
  "DanielL": ["LogicaCp"],
  "OmarLogica": ["LogicaCp"],
  "CarmentL": ["LogicaCp"],
  "ErnestoAnalisis": ["AnalisisCp"],
  "CristinaA": ["AnalisisCp"],
  "MercedesA": ["AnalisisCp"],
  "DalianisAlgebra": ["AlgebraCP"],
  "PepeAL": ["AlgebraCP"],
  "CayetanaAL": ["AlgebraCP"],
  "PacoP": ["ProgramacionCp"],
  "HectorP": ["ProgramacionCp"],
  "CarlaP": ["ProgramacionCp"],
  "JuanPabloCom": ["Programacion","CompilacionCon"],
  "GustavoCom": ["CompilacionCp"],
  "LiaCom": ["CompilacionCp"],
  "DalmauRedesAI": ["AICp","RedesCon"],
  "AntonioRedes": ["RedesCp"],
  "GemaOPT": ["OptimizacionCon"],
```

```
    "DanaOPT": ["OptimizacionCp"],
    "ErnestoOPT": ["OptimizacionCp"],
    "PedroAI": ["AICp"]
  }
}
```

Respuesta

Si la solicitud es exitosa, recibirás un código de estado `201 Created` y un mensaje indicando que la instancia del solucionador se creó con éxito.

Step 3: Agregando Restricciones

Después de crear una instancia de solucionador, puedes agregar varias restricciones a ella. Aquí tienes ejemplos de cómo agregar diferentes tipos de restricciones.

Agregando Restricciones Duras Opcionales

```
POST http://localhost:<port>/solver/add_hard_optional_constraints
Content-Type: application/json

{
    "subjects_name": ["Programacion"],
    "teachers_name": ["Piad"],
    "classrooms_name": ["1"],
    "groups_name": ["C111"],
    "shifts_int": [1],
    "days_int": [1],
    "count_to_be_equals":1
}
```

Agregando Restricciones Duras Verdaderas

```
POST http://localhost:<port>/solver/TrueHardConstraints
Content-Type: application/json

{
  "teachers_name": ["JuanPabloCom"
    ],
  "subjects_name": ["CompilacionCon",
    "CompilacionCp",
    "RedesCon",
    "RedesCp",
    "OptimizacionCon",
    "OptimizacionCp",
    "AICon",
```

```
    "AICp"],
  "classrooms_name": [ "1",
    "2",
    "3",
    "4",
    "5",
    "Postgrado"],
  "groups_name": ["C311"],
  "shifts_int": [1, 2, 3],
  "days_int": [5]
}
```

Agregando Agregando Restricciones Duras Falsas

```
POST http://localhost:<port>/solver/FalseHardConstraints
Content-Type: application/json

{
  "teachers_name": ["Mr. Smith"],
  "subjects_name": ["Math"],
  "classrooms_name": ["101"],
  "groups_name": ["A"],
  "shifts_int": [1],
  "days_int": [1, 2, 3, 4, 5]
}
```

Agregando Restricciones Blandas

```
POST http://localhost:<port>/solver/MaximizeSoftConstraints
Content-Type: application/json

{
  "teachers_name": ["JuanPabloCom",
    "GustavoCom",
    "LiaCom",
    "DalmauRedesAI",
    "AntonioRedes",
    "GemaOPT",
    "DanaOPT",
    "ErnestoOPT",
    "PedroAI"],
  "subjects_name": ["CompilacionCon"
    ],
  "classrooms_name": [ "1",

    "3"
    ],
  "groups_name": ["C311"],
```

```
  "shifts_int": [ 2 ],
  "days_int": [1],
  "alpha_value":50
}
```

```
POST http://localhost:<port>/solver/MinimizeSoftConstraints
Content-Type: application/json

{
  "teachers_name": ["JuanPabloCom",
    "GustavoCom",
    "LiaCom",
    "DalmauRedesAI",
    "AntonioRedes",
    "GemaOPT",
    "DanaOPT",
    "ErnestoOPT",
    "PedroAI"],
  "subjects_name": ["CompilacionCon"
    ],
  "classrooms_name": [ "1",

    "3"
    ],
  "groups_name": ["C311"],
  "shifts_int": [ 2 ],
  "days_int": [1],
  "alpha_value":50
}
```

Step 4: Recuperando el Horario

Una vez que hayas agregado todas las restricciones necesarias, puedes recuperar el horario en varios formatos.

Recuperando el Horario Como un DataFrame

```
GET http://localhost:<port>/dataframe
```

Recuperando el Horario en Formato JSON

```
GET http://localhost:<port>/get_json
```

Descargando el Horario en Formato Excel

```
GET http://localhost:<port>/download_excel
```

Horario Resultante

Lo siguiente es una representación de un horario obtenido al realizar una petición a este endpoint con las particularidades mencionadas anteriormente en el tutorial:

Grupo C111

	1	2	3	4	5
1	Programación CP - Hector P		Algebra CP - Dalianis Algebra		
2		Logica CP - Carmen L	Programación CP - Carla P		
3	Algebra CP - PepeAL		Análisis CP - Mercedes A		

Grupo C112

	1	2	3	4	5
1	Analisis CP - Ernesto Analisis	Programación CP - Carla P			
2	Logica CP - Daniel L	Analisis - Idania	Logica - Yudivian		
3	Algebra CP - Dalianis Algebra	Analisis CP - Mercedes A	Programación - Piad		

Grupo C114

	1	2	3	4	5
1	Analisis CP - Ernesto Analisis	Logica CP - Omar Logica	Logica - Yudivian		
2	Algebra CP - Dalianis Algebra	Analisis CP - Mercedes A	Programación - Piad		
3	Algebra CP - Dalianis Algebra	Programación CP - Carla P	Programación - Piad		

Grupo C115

	1	2	3	4	5
1	Analisis CP - Ernesto Analisis	Logica CP - Carmen L			
2		Programación CP - Paco P			
3			Programación		

Información Adicional

Con el servidor de flask ejecutandose acceder a la dirección `http://localhost:<port>/swagger/` para tener acceso a una interfaz gráfica swagger que brindará información acerca de cada uno de los endpoints así como brindar herramientas para probar el proyecto.

Conclusiones

Este tutorial cubrió los conceptos básicos de usar la API de Tabulación de Tiempo para crear un horario. Siguiendo estos pasos, puedes personalizar el horario de acuerdo a tus necesidades agregando diversas restricciones. Recuerda, la API ofrece flexibilidad en cómo defines tus restricciones, permitiendo una amplia gama de escenarios de programación.